Web development

Web development is the work involved in developing a website for the Internet (World Wide Web) or an intranet (a private network).[1] Web development can range from developing a simple single static page of plain text to complex web applications, electronic businesses, and social network services. A more comprehensive list of tasks to which Web development commonly refers, may include Web engineering, Web design, Web content development, client liaison, client-side/server-side scripting, Web server and network security configuration, and e-commerce development.

Among Web professionals, "Web development" usually refers to the main non-design aspects of building Web sites: writing markup and coding.[2] Web development may use content management systems (CMS) to make content changes easier and available with basic technical skills.

For larger organizations and businesses, Web development teams can consist of hundreds of people (Web developers) and follow standard methods like Agile methodologies while developing Web sites.[1] Smaller organizations may only require a single permanent or contracting developer, or secondary assignment to related job positions such as a graphic designer or information systems technician. Web development may be a collaborative effort between departments rather than the domain of a designated department. There are three kinds of Web developer specialization: front-end developer, back-end developer, and full-stack developer.[3] Front-end developers are responsible for behavior and visuals that run in the user browser, while back-end developers deal with the servers.[4] Since the commercialization of the Web, the industry has boomed and has become one of the most used technologies ever.

Evolution of the World Wide Web and web development

Origin/ Web 1.0

Tim Berners-Lee created the World Wide Web in 1989 at CERN.[5]

The primary goal in the development of the Web was to fulfill the automated information-sharing needs of academics affiliated with institutions and various global organizations. Consequently, HTML was developed in 1993.[6]

Web 1.0 is described as the first paradigm wherein users could only view material and provide a small amount of information.[7] Core protocols of web 1.0 were HTTP, HTML and URI.[8]

Web 2.0

Web 2.0, a term popularised by Dale Dougherty, then vice president of O'Reilly, during a 2004 conference with Media Live, marks a shift in internet usage, emphasizing interactivity.[9][10]

Web 2.0 introduced increased user engagement and communication. It evolved from the static, read-only nature of Web 1.0 and became an integrated network for engagement and communication. It is often referred to as a user-focused, read-write online network.[7]

In the realm of Web 2.0 environments, users now have access to a platform that encourages sharing activities such as creating music, files, images, and movies.[11] The architecture of Web 2.0 is often considered the "backbone of the internet," using standardized XML (Extensible Markup Language) tags to authorize information flow from independent platforms and online databases.[7]

Web 3.0

Web 3.0, considered the third and current version of the web, was introduced in 2014. The concept envisions a complete redesign of the web. Key features include the integration of metadata, precise information delivery, and improved user experiences based on preferences, history, and interests.

Web 3.0 aims to turn the web into a sizable, organized database, providing more functionality than traditional search engines. Users can customize navigation based on their preferences, and the core ideas involve identifying data sources, connecting them for efficiency, and creating user profiles.[7]

This version is sometimes also known as Semantic Web.[12]

Evolution of web development technologies

The journey of web development technologies began with simple HTML pages in the early days of the internet. Over time, advancements led to the incorporation of CSS for styling and JavaScript for interactivity. This evolution transformed static websites into dynamic and responsive platforms, setting the stage for the complex and feature-rich web applications we have today.

- Static HTML Pages (1990s)
- Introduction of CSS (late 1990s)[13]
- JavaScript and Dynamic HTML (1990s - early 2000s)[14][15]

- AJAX (1998)[16]
- Rise of Content management systems (CMS) (mid-2000s)
- Mobile web (late 2000s - 2010s)
- Single-page applications (SPAs) and front-end frameworks (2010s)
- Server-side javaScript (2010s)
- Microservices and API-driven development (2010s - present)
- Progressive web apps (PWAs) (2010s - present)
- JAMstack Architecture (2010s - present)
- WebAssembly (Wasm) (2010s - present)
- Serverless computing (2010s - present)
- AI and machine learning integration (2010s - present)

Web development in future will be driven by advances in browser technology, Web internet infrastructure, protocol standards, software engineering methods, and application trends.[8]

Web development life cycle

The web development life cycle is a method that outlines the stages involved in building websites and web applications. It provides a structured approach, ensuring optimal results throughout the development process.

A typical Web Development process can be divided into 7 steps.

Analysis

Debra Howcraft and John Carroll proposed a methodology in which web development process can be divided into sequential steps. They mentioned different aspects of analysis.[17]

Phase one involves crafting a web strategy and analyzing how a website can effectively achieve its goals. Keil et al.'s research[18] identifies the primary reasons for software project failures as a lack of top management commitment and misunderstandings of system requirements. To mitigate these risks, Phase One establishes strategic goals and objectives, designing a system to fulfill them. The decision to establish a web presence should ideally align with the organization's corporate information strategy.

The analysis phase can be divided into 3 steps:

- Development of a web strategy

- Defining objectives
- Objective analysis

During this phase, the previously outlined objectives and available resources undergo analysis to determine their feasibility. This analysis is divided into six tasks, as follows:

- **Technology analysis:** Identification of all necessary technological components and tools for constructing, hosting, and supporting the site.
- **Information analysis:** Identification of user-required information, whether static (web page) or dynamic (pulled "live" from a database server).
- **Skills analysis:** Identification of the diverse skill sets necessary to complete the project.
- **User analysis:** Identification of all intended users of the site, a more intricate process due to the varied range of users and technologies they may use.
- **Cost analysis:** Estimation of the development cost for the site or an evaluation of what is achievable within a predefined budget.
- **Risk analysis:** Examination of any major risks associated with site development.

Following this analysis, a more refined set of objectives is documented. Objectives that cannot be presently fulfilled are recorded in a Wish List, constituting part of the Objectives Document. This documentation becomes integral to the iterative process during the subsequent cycle of the methodology.[17]

Planning: sitemap and wireframe

It is crucial for web developers to be engaged in formulating a plan and determining the optimal architecture and selecting the frameworks. Additionally, developers/consultants play a role in elucidating the total cost of ownership associated with supporting a website, which may surpass the initial development expenses.

Key aspects in this step are:

- Sitemap creation
- Wireframe creation
- Tech stack

Design and layout

Following the analysis phase, the development process moves on to the design phase, which is guided by the objectives document. Recognizing the incremental growth of websites and the potential lack of good design architecture, the methodology includes iteration to account for

changes and additions over the life of the site. The design phase, which is divided into Information Design and Graphic Design, results in a detailed Design Document that details the structure of the website, database data structures, and CGI scripts.*

The following step, design testing, focuses on early, low-cost testing to identify inconsistencies or flaws in the design. This entails comparing the website's design to the goals and objectives outlined in the first three steps. Phases One and Two involve an iterative loop in which objectives in the Objectives Document are revisited to ensure alignment with the design. Any objectives that are removed are added to the Wish List for future consideration.[17]

Key aspects in this step are:

- Page layouts
- Review
- Approval

Content creation

No matter how visually appealing a website is, good communication with clients is critical. The primary purpose of content production is to create a communication channel through the user interface by delivering relevant information about your firm in an engaging and easily understandable manner. This includes:

- Developing appealing calls to action
- Making creative headlines
- Content formatting for readability
- Carrying out line editing
- Text updating throughout the site development process.

The stage of content production is critical in establishing the branding and marketing of your website or web application. It serves as a platform for defining the purpose and goals of your online presence through compelling and convincing content.

Development

During this critical stage, the website is built while keeping its fundamental goal in mind, paying close attention to all graphic components to assure the establishment of a completely working site.

The procedure begins with the development of the main page, which is followed by the

production of interior pages. The site's navigational structure is being refined in particular.

During this development phase, key functionality such as the Content Management System, interactive contact forms, and shopping carts are activated.

The coding process includes creating all of the site's software and installing it on the appropriate Web servers. This can range from simple things like posting to a Web server to more complex tasks like establishing database connections.

Testing, review and launch

In any web project, the testing phase is incredibly intricate and difficult. Because web apps are frequently designed for a diverse and often unknown user base running in a range of technological environments, their complexity exceeds that of traditional Information Systems (IS). To ensure maximum reach and efficacy, the website must be tested in a variety of contexts and technologies. The website moves to the delivery stage after gaining final approval from the designer. To ensure its preparation for launch, the quality assurance team performs rigorous testing for functionality, compatibility, and performance.

Additional testing is carried out, including integration, stress, scalability, load, resolution, and cross-browser compatibility. When the approval is given, the website is pushed to the server via FTP, completing the development process.

Key aspects in this step are:

- Test Lost Links
- Use code validators
- Check browser

Maintenance and updating

The web development process goes beyond deployment to include a variety of post-deployment tasks.

Websites, in example, are frequently under ongoing maintenance, with new items being uploaded on a daily basis. The maintenance costs increases immensely as the site grows in size. The accuracy of content on a website is critical, demanding continuous monitoring to verify that both information and links, particularly external links, are updated. Adjustments are made in response to user feedback, and regular support and maintenance actions are carried out to maintain the website's long-term effectiveness.[17]

Traditional development methodologies

Debra Howcraft and John Carroll discussed a few traditional web development methodologies in their research paper:[17]

- **Waterfall:** The waterfall methodology comprises a sequence of cascading steps, addressing the development process with minimal iteration between each stage. However, a significant drawback when applying the waterfall methodology to the development of websites (as well as information systems) lies in its rigid structure, lacking iteration beyond adjacent stages. Any methodology used for the development of Web-sites must be flexible enough to cope with change.[17]
- **Structured Systems Analysis and Design Method (SSADM):** Structured Systems Analysis and Design Method (SSADM) is a widely used methodology for systems analysis and design in information systems and software engineering. Although it does not cover the entire lifecycle of a development project, it places a strong emphasis on the stages of analysis and design in the hopes of minimizing later-stage, expensive errors and omissions.[17]
- **Prototyping:** Prototyping is a software development approach in which a preliminary version of a system or application is built to visualize and test its key functionalities. The prototype serves as a tangible representation of the final product, allowing stakeholders, including users and developers, to interact with it and provide feedback.
- **Rapid Application Development:** Rapid Application Development (RAD) is a software development methodology that prioritizes speed and flexibility in the development process. It is designed to produce high-quality systems quickly, primarily through the use of iterative prototyping and the involvement of end-users. RAD aims to reduce the time it takes to develop a system and increase the adaptability to changing requirements.
- **Incremental Prototyping:** Incremental prototyping is a software development approach that combines the principles of prototyping and incremental development. In this methodology, the development process is divided into small increments, with each increment building upon the functionality of the previous one. At the same time, prototypes are created and refined in each increment to better meet user requirements and expectations.

Key technologies in web development

Developing a fundamental knowledge of client-side and server-side dynamics is crucial.

The goal of front-end development is to create a website's user interface and visual components that users may interact with directly. On the other hand, back-end development works with databases, server-side logic, and application functionality. Building reliable and user-friendly

online applications requires a comprehensive approach, which is ensured by collaboration between front-end and back-end engineers.

Front-end development

Front-end development is the process of designing and implementing the user interface (UI) and user experience (UX) of a web application. It involves creating visually appealing and interactive elements that users interact with directly. The primary technologies and concepts associated with front-end development include:

Technologies

The 3 core technologies for front-end development are:

- **HTML (Hypertext Markup Language):** HTML provides the structure and organization of content on a webpage.
- **CSS (Cascading Style Sheet):** Responsible for styling and layout, CSS enhances the presentation of HTML elements, making the application visually appealing.
- **JavaScript:** It is used to add interactions to the web pages. Advancement in JavaScript has given rise to many popular front- end frameworks like React, Angular and Vue.js etc.

User interface design

User experience design focuses on creating interfaces that are intuitive, accessible, and enjoyable for users. It involves understanding user behavior, conducting usability studies, and implementing design principles to enhance the overall satisfaction of users interacting with a website or application. This involves wireframing, prototyping, and implementing design principles to enhance user interaction. Some of the popular tools used for UI Wireframing are -

- Sketch for detailed, vector-based design
- Moqups for beginners
- Figma for a free wireframe app
- UXPin for handing off design documentation to developers
- MockFlow for project organization
- Justinmind for interactive wireframes
- Uizard for AI-assisted wireframing

Another key aspect to keep in mind while designing is Web Accessibility- Web accessibility ensures that digital content is available and usable for people of all abilities. This involves adhering to standards like the Web Content Accessibility Guidelines (WCAG), implementing

features like alternative text for images, and designing with considerations for diverse user needs, including those with disabilities.

Responsive design

It is important to ensure that web applications are accessible and visually appealing across various devices and screen sizes. Responsive design uses CSS media queries and flexible layouts to adapt to different viewing environments.

Front-end frameworks

A framework is a high-level solution for the reuse of software pieces, a step forward in simple library-based reuse that allows for sharing common functions and generic logic of a domain application.[19]

Frameworks and libraries are essential tools that expedite the development process. These tools enhance developer productivity and contribute to the maintainability of large-scale applications. Some popular front-end frameworks are:

- **React:** A JavaScript library for building user interfaces, maintained by Facebook. It allows developers to create reusable UI components.
- **Angular:** A TypeScript-based front-end framework developed and maintained by Google. It provides a comprehensive solution for building dynamic single-page applications.
- **Vue.js:** A progressive JavaScript framework that is approachable yet powerful, making it easy to integrate with other libraries or existing projects.

State management

Managing the state of a web application to ensure data consistency and responsiveness. State management libraries like Redux (for React) or Vuex (for Vue.js) play a crucial role in complex applications.

Back-end development

Back-end development involves building the server-side logic and database components of a web application. It is responsible for processing user requests, managing data, and ensuring the overall functionality of the application. Key aspects of back-end development include:

Server/ cloud instance

An essential component of the architecture of a web application is a server or cloud instance. A cloud instance is a virtual server instance that can be accessed via the Internet and is created, delivered, and hosted on a public or private cloud. It functions as a physical server that may

seamlessly move between various devices with ease or set up several instances on one server. It is therefore very dynamic, scalable, and economical.

Databases

Database management is crucial for storing, retrieving, and managing data in web applications. Various database systems, such as MySQL, PostgreSQL, and MongoDB, play distinct roles in organizing and structuring data. Effective database management ensures the responsiveness and efficiency of data-driven web applications. There are 3 types of databases:

- **Relational databases:** Structured databases that use tables to organize and relate data. Common Examples include - MySQL, PostgreSQL and many more.
- **NoSQL databases:** NoSQL databases are designed to handle unstructured or semi-structured data and can be more flexible than relational databases. They come in various types, such as document-oriented, key-value stores, column-family stores, and graph databases. Examples: MongoDB, Cassandra, ScyllaDB, CouchDB, Redis.
- **Document stores:** Document stores store data in a semi-structured format, typically using JSON or XML documents. Each document can have a different structure, providing flexibility. Examples: MongoDB, CouchDB.
- **Key-value stores:** Key-value stores store data as pairs of keys and values. They are simple and efficient for certain types of operations, like caching. Examples: Redis, DynamoDB.
- **Column-family stores:** Column-family stores organize data into columns instead of rows, making them suitable for large-scale distributed systems and analytical workloads. Examples: Apache Cassandra, HBase.
- **Graph databases:** Graph databases are designed to represent and query data in the form of graphs. They are effective for handling relationships and network-type data. Examples: Neo4j, Amazon Neptune.
- **In-memory databases:** In-memory databases store data in the system's main memory (RAM) rather than on disk. This allows for faster data access and retrieval. Examples: Redis, Memcached.
- **Time-series databases:** Time-series databases are optimized for handling time-stamped data, making them suitable for applications that involve tracking changes over time. Examples: InfluxDB, OpenTSDB.
- **NewSQL databases:** NewSQL databases aim to provide the scalability of NoSQL databases while maintaining the ACID properties (Atomicity, Consistency, Isolation, Durability) of traditional relational databases. Examples: Google Spanner, CockroachDB.

- **Object-oriented databases:** Object-oriented databases store data in the form of objects, which can include both data and methods. They are designed to work seamlessly with object-oriented programming languages. Examples: db4o, ObjectDB.

The choice of a database depends on various factors such as the nature of the data, scalability requirements, performance considerations, and the specific use case of the application being developed. Each type of database has its strengths and weaknesses, and selecting the right one involves considering the specific needs of the project.

Application programming interface (APIs)

Application Programming Interfaces are sets of rules and protocols that allow different software applications to communicate with each other. APIs define the methods and data formats that applications can use to request and exchange information.

- RESTful APIs and GraphQL are common approaches for defining and interacting with web services.

Types of APIs

- **Web APIs:** These are APIs that are accessible over the internet using standard web protocols such as HTTP. RESTful APIs are a common type of web API.
- **Library APIs:** These APIs provide pre-built functions and procedures that developers can use within their code.
- **Operating System APIs:** These APIs allow applications to interact with the underlying operating system, accessing features like file systems, hardware, and system services.

Server-side languages

Programming languages aimed at server execution, as opposed to client browser execution, are known as server-side languages. These programming languages are used in web development to perform operations including data processing, database interaction, and the creation of dynamic content that is delivered to the client's browser. A key element of server-side programming is server-side scripting, which allows the server to react to client requests in real time.

Some popular server-side languages are:

1. **PHP:** PHP is a widely used, open-source server-side scripting language. It is embedded in HTML code and is particularly well-suited for web development.
2. **Python:** Python is a versatile, high-level programming language used for a variety of purposes, including server-side web development. Frameworks like Django and Flask make it easy to build web applications in Python.
3. **Ruby:** Ruby is an object-oriented programming language, and it is commonly used for web development. Ruby on Rails is a popular web framework that simplifies the process of

building web applications.

4. **Java:** Java is a general-purpose, object-oriented programming language. Java-based frameworks like Spring are commonly used for building enterprise-level web applications.
5. **Node.js (JavaScript):** While JavaScript is traditionally a client-side language, Node.js enables developers to run JavaScript on the server side. It is known for its event-driven, non-blocking I/O model, making it suitable for building scalable and high-performance applications.
6. **C# (C Sharp):** C# is a programming language developed by Microsoft and is commonly used in conjunction with the .NET framework for building web applications on the Microsoft stack.
7. **ASP.NET:** ASP.NET is a web framework developed by Microsoft, and it supports languages like C# and VB.NET. It simplifies the process of building dynamic web applications.
8. **Go (Golang):** Go is a statically typed language developed by Google. It is known for its simplicity and efficiency and is increasingly being used for building scalable and high-performance web applications.
9. **Perl:** Perl is a versatile scripting language often used for web development. It is known for its powerful text-processing capabilities.
10. **Swift:** Developed by Apple, Swift is used for server-side development in addition to iOS and macOS app development.
11. **Lua:** Lua is used for some embedded web servers, e.g. the configuration pages on a router, including OpenWRT.

Security measures

Implementing security measures to protect against common vulnerabilities, including SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). Authentication and authorization mechanisms are crucial for securing data and user access.

Testing, debugging and deployment

Thorough testing and debugging processes are essential for identifying and resolving issues in a web application. Testing may include unit testing, integration testing, and user acceptance testing. Debugging involves pinpointing and fixing errors in the code, ensuring the reliability and stability of the application.

- **Unit Testing:** Testing individual components or functions to verify that they work as expected.
- **Integration Testing:** Testing the interactions between different components or modules to ensure they function correctly together.

- **Continuous Integration and Deployment (CI/CD):** CI/CD pipelines automate testing, deployment, and delivery processes, allowing for faster and more reliable releases.

Full-stack development

Full-stack development refers to the practice of designing, building, and maintaining the entire software stack of a web application. This includes both the frontend (client-side) and backend (server-side) components, as well as the database and any other necessary infrastructure. A full-stack developer is someone who has expertise in working with both the frontend and backend technologies, allowing them to handle all aspects of web application development.

- **MEAN** (MongoDB, Express.js, Angular, Node.js) and **MERN** (MongoDB, Express.js, React, Node.js) are popular full-stack development stacks that streamline the development process by providing a cohesive set of technologies.

Web development tools and environments

Efficient web development relies on a set of tools and environments that streamline the coding and collaboration processes:

1. **Integrated development environments (IDEs):** Tools like Visual Studio Code, Atom, and Sublime Text provide features such as code highlighting, autocompletion, and version control integration, enhancing the development experience.
2. **Version control:** Git is a widely used version control system that allows developers to track changes, collaborate seamlessly, and roll back to previous versions if needed.
3. **Collaboration tools:** Communication platforms like Slack, project management tools such as Jira, and collaboration platforms like GitHub facilitate effective teamwork and project management.

Security practices in web development

Security is paramount in web development to protect against cyber threats and ensure the confidentiality and integrity of user data. Best practices include encryption, secure coding practices, regular security audits, and staying informed about the latest security vulnerabilities and patches.

- **Common threats:** Developers must be aware of common security threats, including SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).
- **Secure coding practices:** Adhering to secure coding practices involves input validation, proper data sanitization, and ensuring that sensitive information is stored and transmitted securely.
- **Authentication and authorization:** Implementing robust authentication mechanisms, such as OAuth or JSON Web Tokens (JWT), ensures that only authorized users can access specific

resources within the application.

Agile methodology in web development

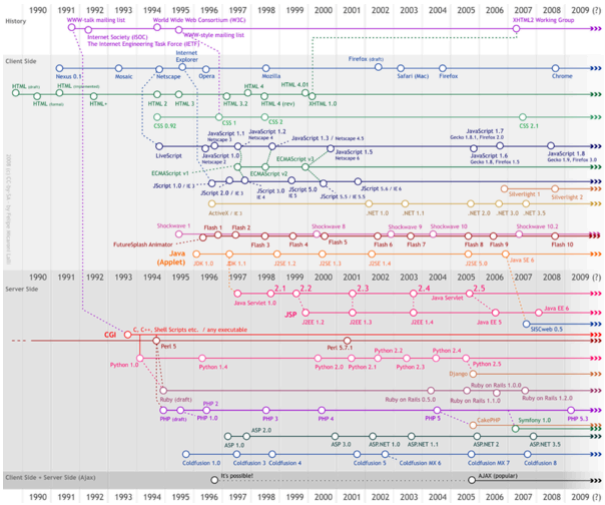
Agile manifesto and principles

Agile is a set of principles and values for software development that prioritize flexibility, collaboration, and customer satisfaction. The four key values are:

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

Agile concepts in web development

1. **Iterative and incremental development:** Building and refining a web application through small, repeatable cycles, enhancing features incrementally with each iteration.
2. **Scrum and kanban:** Employing agile frameworks like Scrum for structured sprints or Kanban for continuous flow to manage tasks and enhance team efficiency.
3. **Cross-functional teams:** Forming collaborative teams with diverse skill sets, ensuring all necessary expertise is present for comprehensive web development.
4. **Customer collaboration:** Engaging customers throughout the development process to gather feedback, validate requirements, and ensure the delivered product aligns with expectations.
5. **Adaptability to change:** Embracing changes in requirements or priorities even late in the development process to enhance the product's responsiveness to evolving needs.
6. **User stories and backlog:** Capturing functional requirements through user stories and maintaining a backlog of prioritized tasks to guide development efforts.
7. **Continuous integration and continuous delivery (CI/CD):** Implementing automated processes to continuously integrate code changes and deliver updated versions, ensuring a streamlined and efficient development pipeline.



See also

- Outline of web design and web development
- Web design
- Web development tools
- Web application development
- Web developer

References

1. "What is Web Development? - Definition from Techopedia" (https://www.techopedia.com/definition/23889/web-development) . *Techopedia.com*. Retrieved 2018-12-07.
2. Campbell, Jennifer (2017). *Web Design: Introductory*. Cengage Learning. p. 27.
3. Northwood, Chris (2018-11-19). *The Full Stack Developer: Your Essential Guide to the Everyday Skills Expected of a Modern Full Stack Web Developer* (https://books.google.com/books?id=vvd6DwAAQBAJ) . Apress. ISBN 978-1-4842-4152-3.
4. "Discover The Difference Between Front-End Vs. Back-End Developer – Forbes Advisor" (https://www.forbes.com/advisor/education/front-end-vs-back-end-developer) . *forbes.com*. May 30, 2023.
5. "A short history of the Web" (https://home.cern/science/computing/birth-web/short-history-web) . *CERN*. 2023-12-04. Retrieved 2023-12-10.
6. "WebD2: A Brief History of HTML" (https://www.washington.edu/accesscomputing/webd2/student/unit1/module3/html_history.html#:~:text=The%20first%20version%20of%20HTML,official%20standard%20in%20December%201999.) . *www.washington.edu*. Retrieved 2023-12-10.

7. Soni, Anuj; Gupta, Sachin; Talwandi, Navjot Singh (September 2023). "Evolution Of Web Technologies in Recent Years" (https://www.jetir.org/papers/JETIR2309347.pdf) (PDF). *Journal of Emerging Technologies and Innovative Research*. **10** (9). ISSN 2349-5162 (https://search.worldcat.org/issn/2349-5162) .
8. Jazayeri, Mehdi (2007). "Some Trends in Web Application Development". *Future of Software Engineering (FOSE '07)* (https://ieeexplore.ieee.org/document/4221621) . pp. 199–213. doi:10.1109/fose.2007.26 (https://doi.org/10.1109%2Ffose.2007.26) . ISBN 978-0-7695-2829-8. S2CID 7279594 (https://api.semanticscholar.org/CorpusID:7279594) . Retrieved 2023-11-30.
9. "Web 2.0" (http://www.paulgraham.com/web20.html) . *www.paulgraham.com*. Retrieved 2023-12-10.
10. "How Web 2.0 Works" (https://computer.howstuffworks.com/web-20.htm) . *HowStuffWorks*. 2007-12-28. Retrieved 2023-12-10.
11. Hall, Heather (2022-05-01). "Web 2.0 Explained: Everything You Need To Know" (https://history-computer.com/web-2-0/) . *History-Computer*. Retrieved 2023-12-10.
12. Du, Xiaofeng; Song, William; Munro, Malcolm (2009), Barry, Chris; Lang, Michael; Wojtkowski, Wita; Conboy, Kieran (eds.), "Semantic Service Description Framework for Address" (http://link.springer.com/10.1007/978-0-387-78578-3_35) , *Information Systems Development*, Boston, MA: Springer US, pp. 1033–1045, doi:10.1007/978-0-387-78578-3_35 (https://doi.org/10.1007%2F978-0-387-78578-3_35) , ISBN 978-0-387-78577-6, retrieved 2023-11-30
13. "20 Years of CSS" (https://www.w3.org/Style/CSS20/) . *W3C*. 2016-12-17. Retrieved 2023-12-10.
14. "JavaScript History" (https://www.w3schools.com/js/js_history.asp) . *www.w3schools.com*. Retrieved 2023-12-10.
15. "A brief history of PHP" (https://ifj.edu.pl/private/krawczyk/php/intro-history.html) . *ifj.edu.pl*. Retrieved 2023-12-10.
16. . 2007-06-23 https://web.archive.org/web/20070623125327/http://www.alexhopmann.com/xmlhttp.htm (https://web.archive.org/web/20070623125327/http://www.alexhopmann.com/xmlhttp.htm) . Archived from the original (http://www.alexhopmann.com/xmlhttp.htm) on 2007-06-23. Retrieved 2023-12-10. `{{cite web}}`: Missing or empty `|title=` (help)
17. Howcroft, Debra; Carroll, John (2000). "A proposed methodology for web development" (https://aisel.aisnet.org/ecis2000/73/) . *Ecis 2000 Proceedings*.

18. Keil, Mark; Cule, Paul E.; Lyytinen, Kalle; Schmidt, Roy C. (November 1998). "A framework for identifying software project risks" (https://doi.org/10.1145%2F287831.287843) . *Communications of the ACM*. **41** (11): 76–83. doi:10.1145/287831.287843 (https://doi.org/10.1145%2F287831.287843) . ISSN 0001-0782 (https://search.worldcat.org/issn/0001-0782) .
19. Salas-Zárate, María del Pilar; Alor-Hernández, Giner; Valencia-García, Rafael; Rodríguez-Mazahua, Lisbeth; Rodríguez-González, Alejandro; López Cuadrado, José Luis (May 2015). "Analyzing best practices on Web development frameworks: The lift approach" (https://doi.org/10.1016%2Fj.scico.2014.12.004) . *Science of Computer Programming*. **102**: 1–19. doi:10.1016/j.scico.2014.12.004 (https://doi.org/10.1016%2Fj.scico.2014.12.004) .